I.J. Wireless and Microwave Technologies, **2025, 2, 51-67**
Published Online on April 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijwmt.2025.02.05

Evaluation of Machine Learning Algorithms for Malware Detection: A Comprehensive Review

Sadia Haq Tamanna
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: tamanna55.cse@gmail.com
ORCID iD: https://orcid.org/0009-0002-8676-4978

Muhammad Muhtasim*
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: muhtasim222@gmail.com
ORCID iD: https://orcid.org/0000-0001-5741-603X
*Corresponding Author

Aroni Saha Prapty
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: aronisahaprapty@gmail.com
ORCID iD: https://orcid.org/0009-0006-9530-8164

Amrin Nahar
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: amrinnahar101@gmail.com
ORCID iD: https://orcid.org/0009-0001-4406-0203

Md. Tanvir Ahmed Tagim
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: tanvirahmedtagim@gmail.com
ORCID iD: https://orcid.org/0009-0000-0809-6228

Fahmida Rahman Moumi
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: fahmidarahmanmoumi@gmail.com
ORCID iD: https://orcid.org/0009-0001-9221-1539

Shadia Afrin
Department of Computer Science and Engineering, Bangladesh Army University of Engineering & Technology, Qadirabad Cantonment, Natore-6431, Bangladesh
E-mail: afrinshadia82@gmail.com
ORCID iD: https://orcid.org/0009-0007-9448-1536

Received: 06 June, 2024; Revised: 26 July, 2024; Accepted: 02 February, 2025; Published: 08 April, 2025

Abstract: Malware outperforms conventional signature-based techniques by posing a dynamic and varied threat to digital environments. In cybersecurity, machine learning has become a potent device, providing flexible and data-driven models for malware identification. The significance of choosing the optimal method for this purpose is emphasized in this review paper. Assembling various datasets comprising benign and malicious samples is the first step in the research process. Important data pretreatment procedures like feature extraction and dimensionality reduction are also included. Machine learning techniques, ranging from decision trees to deep learning models, are evaluated based on metrics like

as accuracy, precision, recall, F1-score, and ROC-AUC, which determine how well they distinguish dangerous software from benign applications. A thorough examination of numerous studies shows that the Random Forest algorithm is the most effective in identifying malware. Because Random Forest can handle complex and dynamic malware so well, it performs very well in batch and real-time scenarios. It also performs exceptionally well in static and dynamic analysis circumstances. This study emphasizes how important machine learning is, and how Random Forest is the basis for creating robust malware detection. Its effectiveness, scalability, and adaptability make it a crucial tool for businesses and individuals looking to protect sensitive data and digital assets. In conclusion, by highlighting the value of machine learning and establishing Random Forest as the best-in-class method for malware detection, this review paper advances the subject of cybersecurity. Ethical and privacy concerns reinforce the necessity for responsible implementation and continuous research to tackle the changing malware landscape.

Index Terms: Malware Detection, Static Analysis, Dynamic Analysis, Android Security, Malware Classification, Random Forest

1. Introduction

Malware is a ubiquitous digital menace that compromises the security of our globalized society. These malicious entities are always evolving to take advantage of network vulnerabilities, ranging from viruses to ransomware and advanced cyber threats. Conventional signature-based defenses find it difficult to adapt to their polymorphic changes. Artificial intelligence's machine learning division proves to be a potent ally in this digital conflict. It makes the distinction between software that is harmful and software that is benign using intricate algorithms and data-driven models. This research aims to transform cybersecurity by determining the optimal machine-learning method for malware detection. We carefully compare different algorithms using measures like recall, accuracy, precision, F1-score, and ROC-AUC. The project aims to improve cybersecurity by providing workable answers to a threat that is always changing. It looks at methods for gathering data,
etting it ready, extracting features, choosing a model, and post-processing. Our goal is to identify the machine learning method that works best in this situation by performing this. The impact of this research is enormous, as it helps individuals and companies strengthen their digital defenses and fight a constantly shifting cyber threat scenario.

The word "malware" refers to a broad category of malicious software that is intended to harm or impair a computer, network, or digital device's ability to operate normally. A variety of programs and files, including ransomware, worms, Trojan horses, spyware, and adware, are categorized as "malware". Every one of these malware classifications has distinct traits and ways of functioning of its own. Virus is known as viruses that attach themselves to other software or files that aren't malicious. The virus multiplies to infect additional files or computers once it is activated by the executed infected file.Worms are independent programs with the ability to multiply and propagate on their own. To spread, they usually take advantage of holes in operating systems or software. Sometimes referred to as Trojan horses, Trojans are malicious programs that impersonate reliable ones. Once installed, they can cause a great deal of damage, such as fire damage, backdoor access, and data theft. Spyware is designed to covertly gather information about a user's online activity, spyware is frequently installed without the user's knowledge or consent. Adware displays unsolicited advertisements to the user, usually in the form of pop-up windows or banners. It may not always be damaging, but it can still cause inconvenience and compromise user privacy. Malicious software known as ransomware encrypts a user's files and demands a fee to release the key. It might have terrible consequences for individuals and organizations. There are several approaches and technologies used in malware detection, each with advantages and disadvantages. The following are important methods to identify malware. Signature-Based Detection is predicated on patterns or signatures of known malware. It works well against known threats but has trouble with malware that is polymorphic, and zero-day exploited. Heuristic analysis uses known signatures as a starting point and searches for questionable behavior patterns. It can identify previously unidentified malware, but it can also produce false positives.Behavioral Analysis looks for abnormalities in the way software behaves. It is a useful tactic since it can recognize polymorphic malware and zero-day exploits. Machine learning classifies and analyzes data using algorithms. It's a flexible method that can recognize malware that hasn't been seen before and adjust to changing threats.
Sandboxing is the process of running dubious code in a safe setting to watch how it behaves. Certain malware may detect and avoid sandboxes, even though it is effective.Anomaly detection is the process of spotting abnormalities in behavior that may point to malware. It works very well with zero-day exploits. Cloud-Based Solutions is to discover and lessen risks, cloud-based malware detection makes use of a network of devices' combined intelligence. It works well for quickly identifying and neutralizing new threats.

Static analysis is a popular method for identifying malware because it involves closely analyzing a program's code and structure without ever running it. The first step is file inspection, wherein source or binary code is carefully examined for any questionable content. Typical methods include resource inspection, code analysis, signature-based detection, decomplication for easier human comprehension, packer detection, and obfuscation detection. Using unique rules, rule-based detection looks for dangerous patterns. Static analysis is fast and effective, but it can overlook

polymorphic or camouflaged malware and produce false positives when known malware behaves like normal software.

A key technique for detecting malware in situations where potentially dangerous software is run in a controlled setting is dynamic analysis. Behavioral analysis, which focuses on runtime behavior and keeps track of file interactions, network activity, system calls, and registry modifications, is the fundamental component of dynamic analysis. Memory analysis examines program memory for odd behavior. By classifying malware according to observed behavior, dynamic analysis detects dangerous actions like file encryption, illegal alterations, or data theft.

2. Literature Review

Almomani et al. [1] have used 16 Convolutional Neural Network (CNN) techniques for vision-based Android malware identification in their investigation. They did away with the necessity for conventional feature extraction by converting the bytecode of Android apps into visual representations. Although the study did not identify the optimal algorithm, the combined performance of these CNNs produced exceptional outcomes that outperformed previous techniques.

Agrawal et al. [2] compared Random Forest, SVM, and Naive Bayes classifiers concerning Android malware detection. Random Forest outperformed the others in accuracy, demonstrating why it is preferred for malware detection. In order to improve efficiency and accuracy, the work proposes a comprehensive malware detection model that combines supervised and unsupervised machine learning. It focuses on these techniques for assessment.

Damaševi˘cius et al. [3] discussed cybersecurity and suggested using ensemble classification to find malware. It outperforms prior methods by combining convolutional and dense neural networks (CNN) in the first stage as the meta-learner in the second stage. Explanatory artificial intelligence (XAI) and additional improvements to improve classification accuracy with bigger malware databases are the focus of future efforts.

Shhadat et al. [4] discussed the increasing hazard of complicated malware as a result of rapid technological improvement is examined. They look at machine learning techniques to improve malware detection, especially Random Forest. The work highlights the potential of Random Forest for multi-class classification by concentrating on feature selection and a variety of techniques.

Khammas et al. [5] discussed ransomware attacks and suggest identifying them through regular pattern mining and static analysis. They discover that a random forest classifier with particular parameter values works better in terms of forecast time and accuracy than an earlier approach based on opcode analysis, suggesting that it may find practical application.

Kedziora et al. [6] focused on machine learning and reverse engineering Java code for Android malware detection. After evaluating a number of classification methods, they concluded that K Nearest Neighbors and Random Forest were the most effective at identifying malware characteristics, highlighting their potential for improved cybersecurity.

Bae et al. [7] discussed the ransomware detection technique based on identifying malware by its file-related operations is put forth by authors. Their method emphasizes the necessity for specific ransomware detection strategies in thwarting cyber threats by utilizing machine learning and API invocation sequences CF-NCF to achieve high accuracy.

Han et al. [8] focused on API calls as features to handle the problem of identifying malicious Android apps utilizing a dataset of both malicious and benign apps. Their use of Support Vector Machines (SVM) in static analysis produced excellent results with a noteworthy recall rate and overall accuracy. They intend to investigate dynamic analysis in subsequent research to improve their approach even more.

Costa et al. [9] offered the purpose of detecting malware on Android for the Malware APK Detection Solution (MADS). It uses a multi-stage analysis with machine learning (ML) techniques and rules to lower false positives and boost detection effectiveness. MADS provides better detection performance, shorter execution times, and lower CPU use than traditional ML models, even when the precise ML algorithms are not stated. The study highlights the use of rules and machine learning in tandem for efficient Android malware identification.

Khoda et al. [10] discuss the vulnerability of machine learning-based malware detection systems on the Industrial Internet of Things (IoT) to adversarial assaults is examined by the author. They suggest two approaches to sample selection for retraining classifiers: one that uses kernel-based learning (KBL) probability measures, and the other that takes malware cluster centers into account. Both methods perform better than random selection; KBL produces a 6% improvement in detection accuracy. The efficacy of the KBL-based selective adversarial retraining approach for improving IoT security solutions is highlighted in this paper.

Aljabri et al. [11] discussed the significance of website security as well as the growing threat posed by rogue URLs in the online world. With a 99.98% accuracy rate, they highlight the effectiveness of Convolutional Neural Networks (CNN) and XGBoost in their machine learning (ML) algorithms for identifying dangerous URLs. CNNs are good at extracting features and identifying patterns, which makes them appropriate for URL analysis. XGBoost is a dependable technique for solving classification problems, which improves online security.

Senanayake et al. [12] looked at the growing risk of malware assaults on Android handsets and highlights how useful machine learning (ML) detection techniques are. According to the study, deep learning (DL) methods—like neural networks—perform better than conventional machine learning (ML) models because they can recognize intricate

malware patterns. The emphasis on DL techniques suggests their choice for Android malware detection, highlighting their accuracy and opening the possibility of more research into reinforcement learning for source code vulnerability identification, even though it does not specify the optimal method.

Gong et al. [13] discussed the problem of malware spreading through app stores for mobile devices and highlights the necessity for effective machine learning (ML)-based remedies. The system performs admirably with excellent precision and recall, indicating its suitability for large-scale malware detection in app marketplaces, even though the precise ML technique is not stated.

Al-Kasassbeh et al. [14] addressed feature selection and machine learning-based malware classification, stressing the significance of recognizing particular traits in Portable Executable (PE) files in order to effectively classify malware. The J48 classification method is a useful tool for malware detection in antivirus software since it is the most effective at differentiating between clean and infected files.

To enhance detection performance, Gupta et al. [15] discussed the difficulties in malware detection and classification and suggested two methods based on big data technology and ensemble learning. With its exceptional accuracy, the Weighted Voting_RACA algorithm stands out and highlights the need of huge data and ensemble learning for virus detection.

For the purpose of detecting IoT malware, Xiao et al. [16] presented the Behavior-Based Deep Learning Framework (BDLF). The BDLF enhances detection precision by using Stacked AutoEncoders (SAEs) to extract high-level features from behavior graphs.

Sharma et al. [17] used machine learning and opcode recurrence frequency to address the problem of identifying metamorphic malware. Numerous feature selection strategies were used, and classifiers such as Random Forest, LMT, J48 Graft, NBT, and REPTree were able to detect malware with almost 100% accuracy. The most notable algorithm for accuracy was the Random Forest one.

Baptista et al. [18] offered a novel method for detecting malware that uses self-organizing incremental neural networks and binary visualization. This approach generates 1024-length feature vectors. With minimal false positives and negatives, the approach appears to be promising in terms of identifying ransomware in.pdf and.doc files. Enhancements to feature extraction are investigated using gabor texture filters and multiresolution analysis, expanding the system's usefulness to cloud-based, portable antivirus programs.

Lee et al. [19] presented a novel method based on entropy analysis to evaluate file homogeneity and identify files infected with ransomware. This machine learning techniques to classify infected files, making it possible to recover original information from backups. The best approach for detecting ransomware in various file formats is thought to be a machine learning model that combines entropy analysis with a strong classification technique.

The framework for Android malware detection presented by Kim et al. [20] used a multimodal deep learning model. By efficiently combining several feature types, this approach increases the accuracy of malware detection. The framework exhibits resilience against obfuscation, good detection accuracy, and efficient model updates. The feature combination and improved accuracy of the multimodal deep learning model make it the best method for detecting Android malware.

Kalakoti et al. [21] discussed minimizing feature sets for machine learning in order to detect IoT botnet attacks at different phases of their life cycle. The study highlights the significance of host-based and channel-based elements and displays effective feature selection. One particularly good technique for rapid classification and high detection rates is Sequential Backward Selection (SBS).

Ayofe Azeez et al. [22] suggested utilizing machine learning classifiers and fully linked one-dimensional convolutional neural networks (CNNs) in a two-stage ensemble learning approach for malware detection. The greatest results were obtained with an ensemble of seven neural networks and ExtraTrees, highlighting end-to-end learning for efficient malware detection. In the future, the study recommends adding explainable AI techniques and investigating unsupervised ensemble learning.

An ensemble learning method for malware detection is described by Kouliaridis et al. [23], using 15 machine learning classifiers for additional assessment after the initial classification of one-dimensional fully connected CNNs. The optimal combination of seven neural networks and ExtraTrees emphasizes end-to-end learning for effective malware detection without the need for manual feature engineering. The framework is recommended for use as a reference for future research on Android malware detection and unsupervised ensemble learning should be investigated.

Cai et al. [24] presented JOWMDroid, a cutting-edge technique that combines feature weighting and significance estimations for Android malware detection. By jointly optimizing weight-mapping functions and classifier parameters using differential evolution (DE), the work considerably improves malware detection accuracy using weight-aware classifiers. DE is essential for maximizing weights and parameters for accurate malware classification, which is how JOWMDroid's performance is optimized.

Khan et al. [25] provided "DNAact-Ran," a ransomware detection method based on digital DNA sequencing design limitations and Kmer frequency vectors. Although the report emphasizes DNAact-Ran's high accuracy, precision, recall, and f-measure, it doesn't identify the machine learning algorithm or explain why it was chosen for the method.

Yili et al. [26] provided a machine learning approach for detecting malicious Domain Generation Algorithms (DGAs) that are employed in cyberattacks. A two-level machine learning model, prediction models, dynamic blacklisting, and feature extraction are all part of the system. High accuracy in DGA detection is demonstrated by the experimental results. It is especially noteworthy that the Deep Neural Network (DNN) model can handle big datasets

and improve classification accuracy.

In order to improve security and privacy, Rajesh Kumar et al. [27] presented a unique architecture for Android IoT malware detection that makes use of blockchain technology in conjunction with a naive Bayes classifier based on decision trees.

Shaukat et al. [28] examined the expanding field of cyber threats by assessing support vector machines, deep belief networks, and decision trees for the detection of malware, spam, and intrusions. The study emphasized that in order to improve cyber threat identification, specialised learning models and a variety of benchmark datasets are required. It was not indicated which algorithm was the highest performer.

In order to detect malware,Vinayakumar et al. [29] thoroughly analysed and contrasted deep learning architectures with conventional machine learning algorithms. Their ScaleMalNet architecture proved the superiority of deep learning in detection and categorization by employing static, dynamic, and image processing-based techniques. The suggested deep learning architectures performed better than conventional MLAs, particularly when it came to malware identification via image processing.

Kumara et al. [30] presented AMMDS, which uses virtual machine introspection (VMI) and memory forensic analysis (MFA) to detect malware. It consists of OFMC, which uses machine learning to classify unknown malware without revealing the algorithm, and OMD, which handles known malware. The system showed promise and had no effect on performance, thus more research into Linux-based defences against sophisticated Linux malware is necessary.

Urooj et al. [31] used AdaBoost, SVM, static features, and huge datasets to address Android malware vulnerability. The study underlined the complexity of the code in Android apps, suggested further research on algorithm intercorrelation, and emphasised the difficulties associated with multicollinearity and static analysis.

VisDroid is an image-based method for classifying Android malware that was first presented by Bakour et al. [32]. Six machine learning classifiers were used, along with a variety of local and global characteristics, with Random Forest and ensemble approaches probably being the most used. These strategies turned out to be very successful.

Abawajy et al. [33] talked about Android malware detection and stressed the importance of feature selection for improving machine learning models. Numerous feature selection strategies were investigated, affecting both execution time and accuracy. Some, such as information gain and chi-squared, showed promise. Even if they are computationally efficient, filter-based techniques may have trouble with multicollinearity. The study emphasises how important features and dataset selection strategies are when choosing an algorithm.

Ali et al. [34] discussed sophisticated malware detection methods and suggested utilising dynamic analysis for N-gram- based machine learning detection. With 98.4% classification accuracy, Logistic Regression showed the best level of performance, mostly due to its thorough feature extraction and data analysis. Future research will involve using deep learning techniques on larger datasets and enlarging the feature set.

Fernando et al. [35] examined the use of deep learning and machine learning for ransomware detection. They emphasised the value of early detection, highlighting the superior performance of Logistic Regression and its applicability to ransomware detection based on machine learning features. The focus of the work was on adversarial machine learning and idea drift.

Yang et al. [36] offered two distinct strategies to mitigate malware cybersecurity concerns. For malware classification, the first introduced ensemble models outperformed solo models. The second showed versatility by classifying malware families using t-SNE and k-means clustering. The robustness of ensemble models in improving classification accuracy led to their preference.

Merabet et al. [37] examined the use of machine learning heuristics for malware detection, placing particular emphasis on feature extraction, selection, and classifiers. Because of deep neural networks' (DNNs') greater feature processing and generalisation skills, they fared better than other methods in identifying good files from bad. For sophisticated anti-malware solutions, DNNs outperformed logistic regression, despite its effectiveness.

Feng et al. [38] discussed Android malware detection and provided a real-time, on-device technology called MobiDroid. Using deep learning models—specifically, deep neural networks—effective real-time Android malware detection on mobile devices was demonstrated. Because of their adaptability, these models are selected for on-device applications.

Yuan et al. [39] used BL-AMD, a broad learning-based detector, to address privacy problems related to on-device Android malware detection. BL-AMD outperforms shallow learning models and approaches the performance of deep learning-based models, in contrast to server-trained detectors, by providing on- and offline updates. It is an effective solution due to its lightweight design, on-device training, and resilience to adversarial attacks.

Rathore et al. [40] focused on deep learning and machine learning methods in their discussion of the increasing malware threat. They achieved promising results by combining supervised and unsupervised learning, showcasing the better performance of Random Forest (RF) models. Prospective research directions involve investigating diverse deep learning techniques, such as long short-term memory networks and recurrent neural networks. Due to how well it performed in this situation, RF was chosen.

Malware detection using Graph Isomorphism Networks (GIN) and Control-Flow Graphs (CFG) was first described by Gao et al. [41]. High AUC and accuracy were achieved by the method, and GIN proved to be useful in managing graph- based data for malware classification.

Deep learning was used by Wong et al. [42] to address malware detection. They used ensembles of Support Vector Machines (SVM) optimized for error correction output coding (ECOC), in conjunction with pre-trained ShuffleNet and

DenseNet-201 models. By maximizing SVM parameters and striking a balance between computation and complexity to achieve better or comparable accuracy across malware datasets, the ECOC-SVM ensemble showed effective classification.

Wang et al. [43] introduced a multi-dimensional kernel feature-based architecture with the primary goal of spotting counterfeit Android apps. They assessed different kernel properties and gave priority to qualities linked to memory and signal, and they discovered that their method performed more accurately than rivals. For Android malware detection, decision tree and neural network classifiers were used due to their accuracy and ability to prevent overfitting problems. Grayscale image-based Android virus detection was first introduced by Ünver et al. [44]. They used a mix of global and local features to extract features from grayscale images. Without identifying the optimal method, a number of machine learning classifiers were trained for high accuracy and efficient processing. The excellent classification accuracy of the model was facilitated by the ensemble technique.

Singh et al. [45] presented a dynamic analysis-based behavior-based malware detection technique in a Cuckoo sandbox. By extracting different parts of the runtime and processing features through text mining and singular value decomposition, they improved the accuracy of the malware classifier using the Adaboost ensemble algorithm. What makes this approach special is that it uses text mining of printable strings to find new attributes and uses API and PSI calls to compute Shannon entropy. When it comes to behavior-based malware identification, this approach works incredibly well.

An ensemble classification-based method for malware detection was presented by Venˇckauskas et al. [46]. It employed a variety of machine learning classifiers in the second stage, with ExtraTrees being the most effective meta-learner, and dense and convolutional neural networks (CNNs) in the first. Several models were integrated in this ensemble architecture to improve malware detection skills, with ExtraTrees showing particular proficiency in this area. The technique has the potential for usefully detecting malware in Windows PE.

MalResLSTM, an architecture for Android malware detection, was presented by Alotaibi [47]. It outperformed previous algorithms by using deep residual long short-term memory (LSTM) networks for malware variation recognition and classification. MalResLSTM was chosen because it effectively detected Android malware by integrating deep-learning techniques to extract complex dependencies and information from APK files.

In response to concerns about Android malware, Khariwal et al. [48] suggested a technique for static malware detection that combines permissions and intents for increased precision.They were able to identify the ideal feature set, which included 17 permissions, 20 intents, and 37 attributes. The best algorithm was Random Forest as it fared better than the others in terms of accuracy and showed how effective it is to use a combination of permissions and intents for detecting malware on Android devices.

A unique approach to Android malware detection based on concurrent permissions and APIs was presented by Odat et al. [49]. They investigated various machine learning techniques and used the FP-growth algorithm for feature extraction. Using Random Forest in conjunction with second-level co-existing permissions, the method effectively classified Android malware with the highest accuracy. This approach fared better than more recent models, showing potential in the identification of Android malware.

Using Opcode sequences, Azmoodeh et al. [50] used deep Eigenspace learning to identify malware in the context of the Internet of Battlefield Things (IoBT). With a high accuracy rate, the method showed strong malware detection and resilience against junk code insertion attempts. Although the precise deep learning model employed was not stated, it was probably designed with Opcode sequence analysis in mind.

In this paper, we have provided a comprehensive review of widely used machine learning techniques to gauge the performance of machine learning techniques to detect some widely known cybercrimes. Our paper provides an analysis that offers a thorough summary of several methods for machine learning-based malware detection. The usefulness of ensemble learning, Random Forest (RF), Convolutional Neural Networks (CNNs), and other algorithms have been emphasized by authors in several papers, highlighting the importance of feature extraction and selection techniques. Together, these research advances the science of malware detection and highlight opportunities for enhanced cybersecurity.

3. Results and Discussion

3.1. Our Findings

Various datasets have been used to train machine learning and deep learning models for malware detection. The following table 1 presents a comparison of different algorithms, including CNN, ResNet50, Random Forest, SVM, Naïve Bayes, and K-means Clustering, highlighting their effectiveness. CNN-based models demonstrate the highest accuracy. A review of 50 research papers reveals advancements in machine learning, deep learning, hybrid models, and real-world challenges in malware detection. These findings emphasize the importance of dataset selection and algorithmic improvements in enhancing cybersecurity defenses.

Table 1. Overview of malware detection algorithms based on different datasets and different circumstances

Reference No	Publication Year	Source Of Dataset	Type of Dataset	Algorithms	Best Algorithm	Accuracy
[1]	2021	Leopard Android	14733 malware apps and 2486 benign apps	Convolutional neural network (CNN), Residual Network (ResNet50)	Convolutional neural network (CNN)	99.40%
[2]	2020	Drebin	268 benign applications and 3526 malicious applications	Random Forest, Support Vector Machine (SVM), Na ïve Bayes, K-means Clustering	Random Forest	99.00%
[3]	2021	ClaMP	2722 malware and 2488 Benign instances and has 69 features	Support Vector Machine (SVM), Decision tree	Decision tree	99.00%
[4]	2020	Benchmark	2081 benign and 91 malicious Android applications.	Random Forest, Decision tree Algorithm, Na ïve Bayes, K-means Clustering	Decision tree	98.02%
[5]	2020	Drebin	1680 samples, the present study utilized a 10-fold cross-validation	Support Vector Machine (SVM), Random Forest, Gradient tree boosting	Gradient Tree Boosting	98.25%
[6]	2019	ANDROID JAVA CODE	100, number of features: 696.	Random forest, K- Nearest Neighbors (KNN) Na ïve Bayes, Logistic Regression, Support Vector Machine (SVM)	Random Forest	80.66%
[7]	2019	Drebin	1000 ransomware files, 900 malware files, and 300 benign executable files	Random Forest, Support Vector Machine (SVM)	Random Forest	98.65%
[8]	2020	Malicious Android applications	58,602 Android applications as well as 133,227 features	Random Forest, K- Nearest Neighbors (KNN)	Random Forest	99.51%
[9]	2023	CIC-AndMal2017 CICMalDroid 2020 and R-PackDroid	Samples obtained 1,052 after processing defined 736	Random forest	Random Forest	99.95%
[10]	2019	Androguard tool	3000 malware and 5000 benign applications were selected	Random forest, Na ïve Bayes	Random Forest	73.83%
[11]	2022	PhishTank and Alexa	45 different dataset sources were used.	Convolutional Neural Network (CNN), Extreme Gradient Boosting (XGBoost),	Convolutional Neural Network	99.98%
[12]	2021	Drebin	18 363 malware applications and 217 619 benign applications	Machine Learning (ML), Deep Learning (DL), Support Vector Machine (SVM), Random Forest(RF), K-Means, Neural Network, Convolutional Neural Network(CNN)	ML (Machine Learning) Algorithm	98.87%
[13]	2021	T-Market.	A total number of 426 key APIs as features	Support Vector Machine (SVM), Random Forest(RF), Na ïve Bayes(NB), K-Nearest Neighbors (KNN),Logistic Regression(LR), Gradient Boosted Regression Trees(GBDT), Artificial Neural Network(ANN), Deep Neural Network(DNN)	RF (Random Forest)	96.80%
[14]	2020	Anubis, Kingsoft, ESET NOD32.	1156 malware files in addition to 984 benign files of different formats and 20000 malware instances.	Similar Nearest Neighbours (SNN), Data Mining Algorithm, Decision Tree, Genetic Algorithm	Similar Nearest Neighbours (SNN)	98.90%

[15]	2020	VirusShare, Nothink, VXHeaven, Windows XP, 7, 10 installations directory	100,200 malicious and 98,150 benign files.	Decision Tree (DT), Support Vector Machine (SVM), Random Forest (RF), Na ïve Bayes (NB), K-Nearest Neighbors (KNN)	RF (Random Forest)	99.50%
[16]	2019	VX Heaven.	1760 samples, where 880 are malware samples, and the other 880 are benign	Decision Tree (DT), SAE-DT, SAE-KNN, SAE-NB, SAE-SVM	SAE-NB and SAE-SVM	99.90%
[17]	2019	Kaggle Microsoft College's lab	6010 Malware and 4573 benign files.	Random Forest (RF), Logistic Model Tree (LMT), J48 Graft, REPTREE	REPTRE (Reduced Error Pruning Tree)	99.96%
[18]	2019	VirusShare website	2K benign files and a total of 2K malicious files	Machine Learning (ML), Deep Learning (DL), Support Vector Machine (SVM), Random Forest(RF), K-Means, Neural Network, Convolutional Neural Network(CNN), Na ïve Bayes(NB)	NB (Na ïve Bayes)	73.70%
[19]	2019	Ransomware	6-type data set and also a total of 1,200 files consisting of 100 files for each file format and 100 encrypted files for each file format.	Machine learning models include K- Nearest Neighbors (KNN), linear models, decision trees, decision tree ensemble, kernel trick, and deep learning.	KNN (K-Nearest Neighbors)	99%
[20]	2018	VirusShare, Malgenome project, VirusShare, Google Play App Store	20,000 malware samples and 20,000 benign samples	Support Vector Machine (SVM), Convoluational Neural Network (CNN), Mobile Neural Network (MNN), Deep Neural Network (DNN)	SVM (Support Vector Machine)	80%
[21]	2022	N-BaIoT and MedBIoT	20,000 sample classification 353 type	Random Forest tree,K-Nearest Neighbors (KNN),Decision Tree	Random forest tree	98.93%
[22]	2021	XGBoost	19611 malicias samples	Random Forest tree, Na ïve Bayes, Decision Tree, AdaBoost, Gradient Boosting	Decision Tree	99.24%
[23]	2021	VirusShare or AndroZoo, Drebin and AMD,Drebin [20], Contagio mobile, MalGenome	1000 apps,300 important features	Random Forest, Na ïve Bayes, K-Nearest Neighbors (KNN) Support Vector Machine (SVM) Gradient Boosting	Random forest tree	98.06%
[24]	2020	Drebin,AMD	Drebin contains 5,560 malicious applications 179 different malware and 123,453 benign applications. Second, AMD contains 24,553 samples, categorized 135 varieties among 71 malware	Random Forest Tree Multilayer Perceptron (MLP) K-Nearest Neighbors (KNN) Support Vector Machine (SVM) Logistic Regression	Random forest tree	0.98%

[25]	2020	Drebin	1524 records and 30970 features of which 582are ransomware and 942 are good, dataset, 300 records and 16383 features with 150 ransomware and 150 goodware	Random Forest Na ïve Bayes Sequential Minimal Optimization	Random forest tree	92.10%
[26]	2019	Bambenek Consulting g, malware origins, DGA schema	27 DNS features	Random Forest Tree domain generation algorithm (DGA) Cryptolocker Gradiant Boosting	Gradient boosting	98.06%
[27]	2019	Android IoT Devices Using Various Features	18 363 malware applications and 217 619 benign applications, includes 6192 benign and 5560 malware apps	K-Nearest Neighbors (KNN) Support Vector Machine (SVM) Na ïve Bayes Deep Belief Networks (DBN)	Support Vector Machine (SVM)	95.00%
[28]	2020	Benchmark	444 benign applications and 870 malicious applications	Decision Tree, DBN	Decision Tree	99.69%
[29]	2019	Malimg,m VirusSign and VirusShare	9,339 malware samples from 25 various malware families, About 15,512 malware samples and antivirus	Deep neural networks (DNN), Convoulation Neural Network (CNN)	Deep neural networks (DNN)	98.90%
[30]	2018	VX Heaven, Windows Monitored VM	3375 Windows malware samples,675	Random Forest Tree Na ïve Bayes Logistic Regression	Random forest tree	95.75%
[31]	2022	MalDroid DefenseDroid which include around 14,000 malware samples	56,000+ features	Naive Bayes (NB), AdaBoost (AB)	AdaBoost	93.61%
[32]	2020	Manifest or the Manifest- ARSC-DEX Native_Jar image datase	90% of the dataset has been used for training and 10% for testing the classifiers	Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF)	Random Forest	98.2%
[33]	2021	Google Play Store, VirusShare, Drebin, AndroZoo	11,690 (6190 benign, 5500 malware)	Random Forest	Random Forest	97.64%
[34]	2020	Google, SNDBOX, VTI Source	60 malicious and 60 benign samples	Logistic Regression (LR), Random Forest (RF), Decision Tree (DT), Naive Bayes (NB)	Naive Bayes	98.4%
[35]	2020	WEKA	574 ransomware samples and 442 benign files.	Gradient Boosting Trees (GTB), Multi-Layer Perceptron (MLP)	Multi-layer perceptron	84.086%
[36]	2019	DataCon	30,000 sample data, including 10,000 black samples and 20,000 white samples	Naive Bayes (NB), Random Forest (RF), K-Nearest Neighbors (KNN), Gradient Boosted Decision Trees (GBDT), Support Vector Machine (SVM) Extreme Gradient Boosting (XGBoost)	Random Forest	98.39%
[37]	2019	Drebin	84 features, The dataset had 19611 malacia samples	Random forest, Self-Organizing Feature Maps (SOFMs)	Random Forest	93.21%
[38]	2019	Drebin	29,010 malicious samples contain 5,560 apps	Support Vector Machine (SVM)	Support Vector Machine (SVM)	97%
[39]	2019	Tencent myapp and AndroZoo	55,873 samples	Support Vector Convolutional Neural Network (CNN) AdaBoost	Convolutional Neural Network (CNN)	89.75%
[40]	2019	Malicia and Windows	11, 688 malware and 2, 819 benign executable files	Random Forest (RF)	Random Forest	99.78%

[41]	2022	BODMAS, MGD-BINARY	1000 samples, including 500 malicious samples and 500 benign samples,2381 features,1000 graph data	C-Support Vector Classification, Logistic Regression, Multi-layer perceptron	Multi-layer perceptron	99.16%
[42]	2021	Malimg, MaleVis, virus-MNIST, and Dumpware10	13,760 samples	Deep Random Forest Paradigm, Convolution Neural Network Optimal Error Correction Output Coding (ECOC) ensemble configuration of Support Vector Machines (ECOC-SVM)	Output Coding (ECOC) ensemble configuration of Support Vector Machines (ECOC-SVM)	96.62%
[43]	2021	Android official markets	1275 representative Android malicious apps and 1275 Android benign instances,112 features	Naive Bayes, Decision Tree, Neural Network and K-Nearest Neighbors	Neural Network	97.4%
[44]	2020	Drebin and Malgenom	9700 instances include 4850 benign images and 4850 malicious images	Random forest, k- nearest neighbors, Decision Tree, Bag ging, AdaBoost, and Gradient Boost	AdaBoost	98.75%
[45]	2020	VirusShare, VirusTotal	8524 malware and 7239 malware instances,1500 features	Random Forest (RF), Decision Tree (DT), Support Vector machine (SVM), K-Nearest Neighbors (KNN), Na ïve Bayes (NB), Gradient Boosting and ADA Boosting	AdaBoost	99.54%
[46]	2021	ClaMP	2722 malware apps and 2488 Benign apps	AdaBoost, Decision Tree, ExtraTrees	Extra Trees	99.99%
[47]	2019	Drebin	123,453 benign applications and 5560 malware instances	Deep Residual LSTM, AdaBoost	Deep Residual LSTM	99.32%
[48]	2020	Genome, Drebin, and Koodous	1714 malware apps and 1414 benign instances, 815 intents, and 280 features	Support Vector machine (SVM), Random Forest & Naive Bayes	Random Forest	94.73%
[49]	2023	Drebin, Malgenome And CIC_MALDROID2020	5,550 applications and 9477 benign applications	Random Forest (RF), Decision Tree (J48), and Support Vector Machine (SVM	Random Forest	98%
[50]	2018	VirusTotal	1078 benign and 128 malware instances	Operational Code (OpCode) sequence, Graph Embedding	OpCode sequence	99.68%

3.2. Result Analysis

After analyzing all the papers, we have come across that these papers use the 14 algorithms that are given below in the table. Our study highlights Random Forest (RF) as the most effective algorithm for malware detection based on an analysis of 50 research papers which is shown in table 2. The dataset sources range from Drebin, VirusShare, and Malgenome to real-world Android and Windows environments, covering diverse malware and benign instances. The data table demonstrates that Random Forest consistently achieves high accuracy across various datasets, often exceeding 95% and reaching up to 99.95% in some cases. This suggests that RF is robust in handling different malware detection tasks. Despite this, the review does not deeply analyze the context in which other algorithms might outperform RF. For example, Convolutional Neural Networks (CNN) and Deep Learning techniques exhibit superior performance in specific scenarios, particularly when dealing with large-scale image-based malware classification. Similarly, Na ïve Bayes, Support Vector Machines (SVM), and Gradient Boosting show competitive results, often exceeding 90% accuracy. However, their effectiveness may depend on the dataset size, feature set, or computational constraints. The dominance of RF in the table is likely due to its ability to handle high-dimensional data, its resistance to overfitting, and its ensemble nature, which improves generalization. Nonetheless, limitations such as high computational cost and susceptibility to adversarial attacks are not thoroughly discussed. A more detailed exploration of algorithm-specific strengths and weaknesses would provide a clearer understanding of the trade-offs in malware detection.

Table 2. Calculation of the number of occurrences of the algorithms

Algorithms	Reference No.	No. of occurrences in papers
Random Forest Tree	[2,6,7,8,9,10,13,15,21,23,24, 25,30,33,36,37,40,48,49]	25
Decision Tree	[3,4,22,28]	4
CNN	[1,11,39]	3
SVM	[20,27,38]	3
AdaBoost	[31,44,45]	3
NB	[18,34]	2
Gradient Tree Boosting	[5,26]	2
MLP	[35,41]	2
SNN	[14]	1
ECOC SVM	[42]	1
Extra Trees	[46]	1
Opcode Sequence	[50]	1
Deep Residual LSTM	[47]	1
KNN	[19]	1
Total		**50**

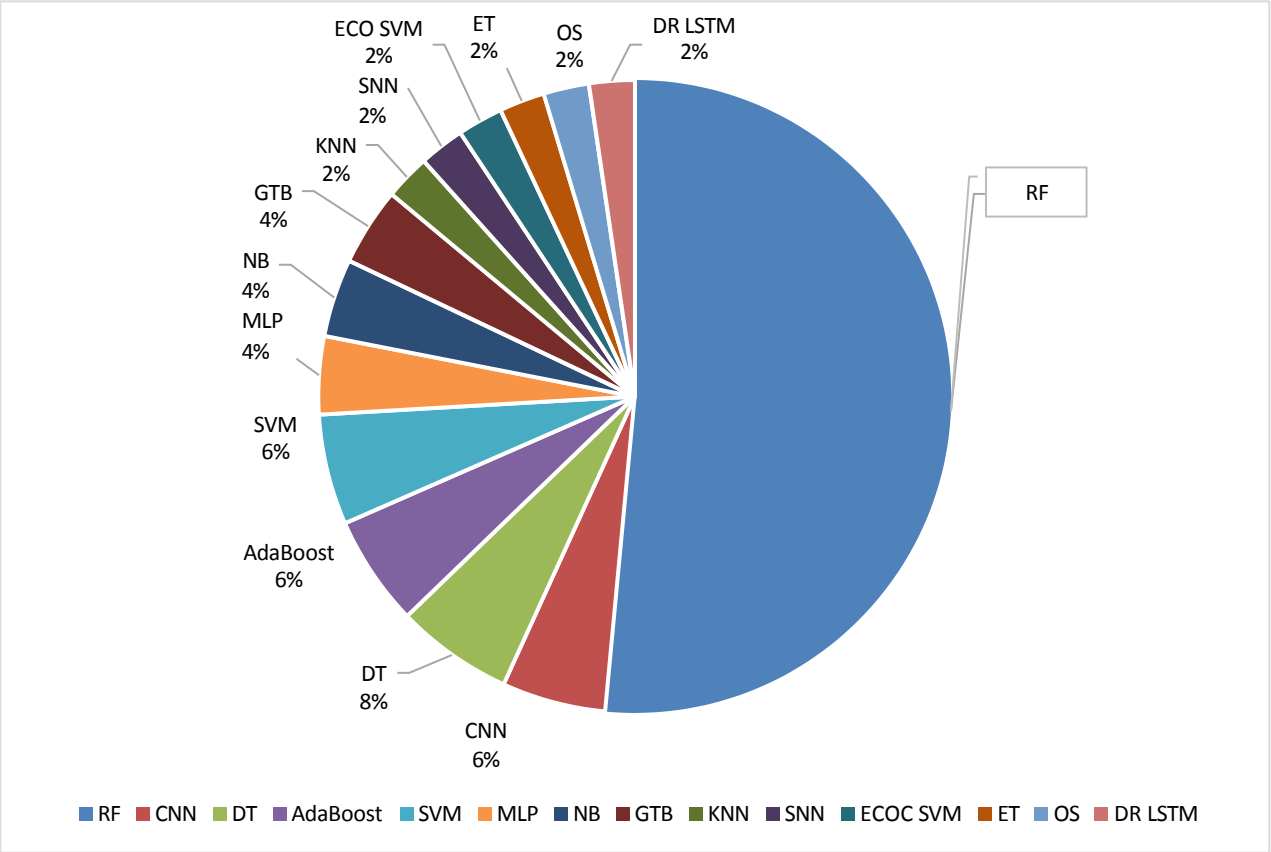

Fig. 1. Pie Chart of the no of occurrences of the algorithms

Fig.1 shows that the "Random Forest" method seems to be the best option based on the maximum number of occurrences (i.e., the number of times each algorithm appears as the top performance in different datasets), as it is the top performer in most datasets. In 25 of the 50 papers, "Random Forest" is the best-performing algorithm. This decision is made solely based on frequency and disregards other elements that can be crucial for choosing an algorithm in particular applications, such as dataset size, computational resources, or interpretability.

Our study primarily summarizes existing literature on malware detection methods, highlighting different datasets, algorithms, and accuracy levels. However, it lacks a significant contribution in terms of proposing new techniques or improvements. The findings mostly reiterate previous research without introducing novel advancements in malware detection. From the dataset table, it is evident that Random Forest, Support Vector Machine (SVM), Decision Trees, and Neural Networks are widely used for malware detection. However, while these techniques yield high accuracy, there is potential for improvement by integrating hybrid models or deep learning architectures to enhance detection efficiency. Some papers, such as [46] and [47], have explored newer techniques like Extra Trees and Deep Residual LSTM, but they remain underutilized. Furthermore, studies such as [11] and [45] highlight the effectiveness of

Convolutional Neural Networks (CNN) and AdaBoost, respectively, suggesting the need for ensemble-based approaches.

A promising direction for improvement is the development of a Hybrid Deep Learning-Based Ensemble Model that integrates CNN, Long Short-Term Memory (LSTM), and Gradient Boosting. CNN can effectively analyze patterns within application code, while LSTM can capture sequential relationships in behavior analysis. Combining these with Gradient Boosting will improve the model's ability to classify malware more accurately and generalize better across diverse datasets. Additionally, Federated Learning could be employed to enhance privacy-preserving malware detection by training models across multiple devices without sharing raw data. To advance malware detection, this review should explore hybrid approaches such as CNN-LSTM with Attention Mechanisms or Transformer-based Malware Classification Models, which have the potential to outperform conventional machine learning techniques. These methods can significantly enhance accuracy, adaptability, and robustness in identifying sophisticated malware variants.

In our study, accuracy is often the most cited performance metric, as seen in overall finding result, which summarizes results from 50 studies on machine learning-based malware detection. Accuracy provides an overall measure of correct classifications, combining both true positives (malware correctly identified) and true negatives (benign applications correctly classified). High accuracy values, such as 99.95% in [9] and 99.98% in [11], suggest strong performance. However, focusing solely on accuracy without considering other evaluation metrics can be misleading, especially when dealing with imbalanced datasets. Many malware datasets contain a significantly higher number of benign applications than malicious ones. In such cases, a model that classifies most instances as benign can achieve high accuracy despite failing to detect malware effectively. For example, in [10], the accuracy is 73.83%, but without precision and recall metrics, it is unclear if false negatives are disproportionately high. To address this, researchers should also consider precision, which measures how many of the detected malware instances are truly malicious, and recall, which indicates how many of the actual malware instances were successfully identified. F1-score, which balances precision and recall, is crucial for evaluating malware detection systems, particularly in real-world cybersecurity applications. Studies using deep learning, such as [29] and [47], demonstrate high accuracy, but it remains unclear how they perform on precision and recall. A model with an accuracy of 99% but a low recall would be ineffective because undetected malware poses a severe security risk. Moreover, false positives (benign applications incorrectly classified as malware) can lead to unnecessary software restrictions, impacting user experience. False negatives (missed malware instances) can be even more dangerous, allowing malware to bypass detection. Therefore, in addition to accuracy, reporting precision, recall, and F1-score ensures a more comprehensive evaluation of model effectiveness. Future studies should adopt a multi-metric evaluation approach to provide a clearer understanding of malware detection performance.

Table 3. The algorithms and their best-achieved accuracy in the papers:

SL No	Algorithms	Accuracy
1	Convolutional Neural Network (CNN)	89.75%
2	Random Forest (RF)	99.78%
3	Decision Tree (DT)	96.5%
4	Naïve Bayes (NB)	98.4%
5	K-Nearest Neighbors (KNN)	96%
6	Support Vector Machine (SVM)	95%
7	AdaBoost	93.61%
8	Multi-Layer Perceptron (MLP)	98%
9	Gradient Tree Boosting (GTB)	97.25%
10	Similar Nearest Neighbors (SNN)	96.9%
11	Output Coding (ECOC) ensemble configuration of Support Vector Machines (ECOC-SVM)	96.62%
12	Extra Trees	94.25%
13	OpCode sequence	97.68%
14	Deep Residual LSTM	95.32%

The dataset sources vary significantly, ranging from Drebin and VirusShare to Google Play Store and Malimg, indicating that different datasets influence the performance of machine learning models which show in table 3. The most commonly used algorithms include Random Forest, Support Vector Machine (SVM), Decision Tree, Naïve Bayes, and Convolutional Neural Networks (CNN). Random Forest emerges as the most frequently used algorithm, demonstrating consistently high accuracy, often exceeding 95%, making it a strong choice for malware classification due to its robustness against overfitting and capability to handle large feature sets. Deep learning techniques, such as CNN and Deep Neural Networks (DNN), have also proven highly effective, particularly in image-based malware detection, with CNN achieving an accuracy as high as 99.98%. However, deep learning models often require extensive computational resources and large datasets, which can be a limitation in real-world applications. Conversely, traditional machine learning models like Decision Trees and Naïve Bayes show relatively strong performance but may struggle with complex malware patterns. Gradient Boosting and AdaBoost have also performed well, with AdaBoost reaching up to 99.54% accuracy. These boosting techniques improve classification performance by combining multiple weak

learners.

However, they can be computationally expensive. In contrast, algorithms like K-Nearest Neighbors (KNN) have shown relatively lower accuracy in some cases, highlighting its sensitivity to dataset size and feature dimensionality. The effectiveness of these algorithms is highly dependent on the dataset characteristics, such as the number of benign versus malicious samples and the number of extracted features. For example, the ClaMP dataset achieved 99.99% accuracy using Extra Trees, suggesting that certain feature-rich datasets can significantly enhance classification performance.

To improve the discussion, the paper should analyze the trade-offs between accuracy, computational cost, interpretability, and real-world applicability. Additionally, it should explore hybrid approaches and ensemble methods, which have been increasingly used to maximize detection rates while minimizing false positives. Addressing these aspects will provide a more comprehensive and critical review of malware detection algorithms, making the study more insightful and aligned with the reviewer's expectations.

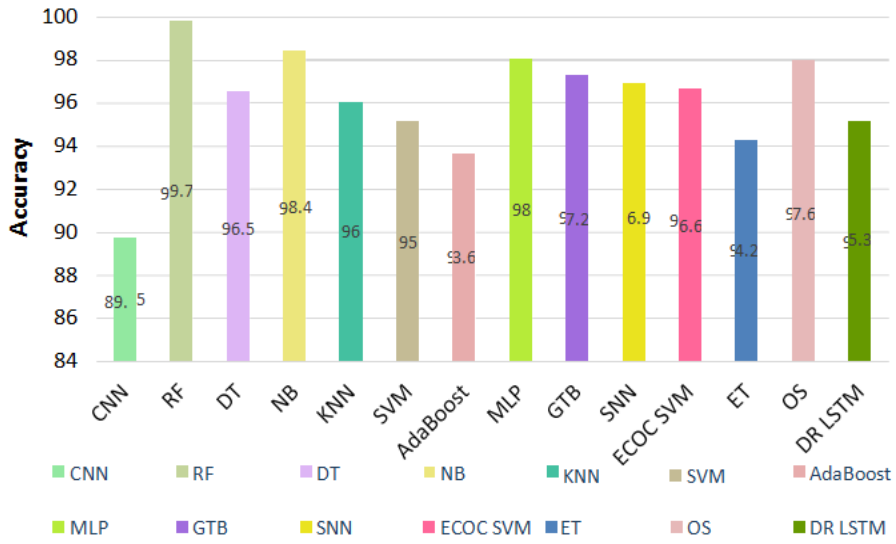

Fig. 2. Bar Chart of the best-recorded accuracy among the algorithms

Fig. 2 shows that Random Forest has the highest accuracy among all of the 14 algorithms. So we suggest this Random Forest algorithm as the best classifier for malware detection based on its maximum occurrences and highest accuracy rate. Feature selection is critical in malware detection, and the manuscript should elaborate on techniques like Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA) in reducing model complexity and preventing overfitting. A more updated discussion should incorporate state-of-the-art methodologies such as reinforcement learning for adaptive threat detection and self-supervised learning for feature extraction in unlabeled datasets. Data preprocessing techniques should be explicitly described, including normalization, feature engineering, and dimensionality reduction, with an analysis of their impact on detection accuracy. Handling imbalanced datasets is a significant challenge, and the paper should discuss methods such as Synthetic Minority Over-sampling Technique (SMOTE) or cost-sensitive learning to improve model fairness. A deeper comparative analysis is necessary to critically evaluate the performance of different machine learning algorithms presented in the data table. The review should highlight discrepancies in accuracy rates across datasets, discussing the influence of dataset composition, feature selection, and algorithmic limitations. The high accuracy of models like CNN (99.40%) and Decision Trees (99.00%) should be examined in the context of dataset characteristics, such as sample size and malware diversity. Additionally, the paper should explore why algorithms like Naïve Bayes underperform (73.70%), emphasizing their limitations in high-dimensional feature spaces. Moreover, studies using ensemble methods like Random Forest (99.95%) and Gradient Boosting (98.25%) demonstrate strong results, warranting a discussion on their ability to handle non-linear patterns in malware classification. The impact of deep learning models such as LSTM (99.32%) and CNN should also be reviewed, especially in handling large-scale malware datasets. The paper should acknowledge the trade-offs between computational cost and performance, as deep models may require significant resources. Furthermore, real-world implementation challenges, such as false positive rates and model explainability, should be explored. The importance of cross-validation techniques, as seen in studies employing 10-fold cross-validation, should be discussed to ensure generalizability. The review should also address potential biases in dataset sources, as malware samples from specific repositories (e.g., Drebin, VirusShare) may not represent emerging threats. Finally, a more critical evaluation of prior studies, focusing on conflicting results and unexplored research gaps, will strengthen the manuscript's contribution to malware detection research.

4. Limitation

These studies have various drawbacks. One study has a short dataset, which results in lower detection rates. It also lacks dynamic analysis and only uses Manifest file analysis. Another has trouble getting consistent results from different machine learning classifiers. Furthermore, a study with a small feature set might not detect malware to the fullest extent possible. Due to imbalanced datasets, classifier bias toward the majority class may occasionally cause minority classes to be incorrectly classified. There are unknowns regarding the length of the analysis and mimicked device limitations when sending huge apps to the cloud for examination. Finally, managing several features in certain ways requires a lot of time and resources. All of these restrictions highlight how difficult and complicated Android malware detection is.

5. Future Scope

Future malware detection research will cover a wide range of topics. First, adding additional benign samples to datasets is essential for gaining a thorough understanding of malware behaviors. Additionally, while computationally expensive, the pursuit of simultaneous feature selection and parameter tweaking holds the possibility of improving detection accuracy. Utilizing other previously trained deep learning networks to assess a feature's appropriateness for transfer learning may present new opportunities. Improvements to evaluation outcomes and graph extraction speed for PE files are still top priorities. To cut down on wait times and increase efficiency, it is crucial to optimize the kernel features dimension through simultaneous in-memory categorization. Precision in Android malware detection is ensured by research into linear and nonlinear algorithms. Additionally, research should concentrate on using image-based object detection approaches to identify camouflaged malware samples, particularly those that use obfuscation techniques. The investigation of the attention mechanism and the bidirectional LSTM can improve detection abilities. It is crucial to assess the framework's vulnerability using adversarial examples. Additionally, evaluating the proposed method against cutting-edge models on larger and broader datasets will shed light on its efficacy. Important steps for a practical application include implementing a prototype in actual IoT and IoBT systems and utilizing distributed computing in a network of IoT nodes. Together, these next projects hope to increase malware detection research and strengthen cybersecurity safeguards.

6. Conclusion

Malware is constantly evolving and posing serious risks to digital systems around the globe, making cybersecurity a dynamic battlefield. New and sophisticated malware strains are difficult for conventional signature-based detection technique to keep up with. Because machine learning makes it possible to create flexible, data-driven models that recognize malware based on its structural and behavioral characteristics, it has become a potent ally in this fight. Diverse and high-quality datasets are essential for efficient machine learning algorithm training. This paper investigates several machine learning methods and metrics that are used to assess how well they detect malware. The most efficient algorithm is Random Forest, which has durability and versatility. But in this fast-paced profession, ethics and continuous innovation are crucial, necessitating cooperation between experts, cybersecurity specialists, and lawmakers to guarantee a safe digital environment. Malware, in the constantly changing field of cybersecurity, continues to be a global danger to digital systems. This dynamic field, which can take many different forms, from conventional viruses to extremely complex ransomware and botnets, poses major hazards to people, businesses, and key infrastructures. Traditional signature-based detection algorithms perform well against known threats, but they struggle to keep up with the rapid creation and dissemination of new malware variants. Machine learning techniques have revolutionized the field of malware detection by enabling the creation of adaptable, data-driven models that identify malware not by its predefined signatures but by its structural and behavioral properties. The area of machine learning-based malware detection is explored in this paper.

References

[1] I. Almomani, A. Alkhayer, and W. El-Shafai, "An Automated Vision-Based Deep Learning Model for Efficient Detection of Android Malware Attacks," *IEEE Access*, vol. 10, pp. 2700–2720, 2022, doi: 10.1109/ACCESS.2022.3140341.

[2] P. Agrawal and B. Trivedi, "Machine Learning Classifiers for Android Malware Detection," in *Advances in Intelligent Systems and Computing*, Springer, 2021, pp. 311–322. doi: 10.1007/978-981-15-5616-6_22.

[3] R. Damaševičius, A. Venčkauskas, J. Toldinas, and Š. Grigaliūnas, "Ensemble ‐ based classification using neural networks and machine learning models for windows pe malware detection," *Electronics (Switzerland)*, vol. 10, no. 4, pp. 1–26, Feb. 2021, doi: 10.3390/electronics10040485.

[4] I. Shhadat, B. Bataineh, A. Hayajneh, and Z. A. Al-Sharif, "The Use of Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware," in *Procedia Computer Science*, Elsevier B.V., 2020, pp. 917–922. doi: 10.1016/j.procs.2020.03.110.
[5] B. M. Khammas, "Ransomware Detection using Random Forest Technique," *ICT Express*, vol. 6, no. 4, pp. 325– 331, Dec. 2020, doi: 10.1016/j.icte.2020.11.001.
[6] M. Kedziora, P. Gawin, M. Szczepanik, and I. Jozwiak, "Malware Detection Using Machine Learning Algorithms and Reverse Engineering of Android Java Code," *International Journal of Network Security & Its Applications*, vol. 11, no. 01, pp. 01–14, Jan. 2019, doi: 10.5121/ijnsa.2019.11101.
[7] S. Il Bae, G. Bin Lee, and E. G. Im, "Ransomware detection using machine learning algorithms," in *Concurrency and Computation: Practice and Experience*, John Wiley and Sons Ltd, Sep. 2020. doi: 10.1002/cpe.5422.
[8] H. Han, S. Lim, K. Suh, S. Park, S. J. Cho, and M. Park, "Enhanced android malware detection: An SVM-based machine learning approach," in *Proceedings - 2020 IEEE International Conference on Big Data and Smart Computing, BigComp 2020*, Institute of Electrical and Electronics Engineers Inc., Feb. 2020, pp. 75–81. doi: 10.1109/BigComp48618.2020.00-96.
[9] L. Da Costa and V. Moia, "A Lightweight and Multi-Stage Approach for Android Malware Detection Using Non- Invasive Machine Learning Techniques," *IEEE Access*, vol. 11, pp. 73127–73144, 2023, doi: 10.1109/ACCESS.2023.3296606.
[10] M. E. Khoda, T. Imam, J. Kamruzzaman, I. Gondal, and A. Rahman, "Robust Malware Defense in Industrial IoT Applications Using Machine Learning with Selective Adversarial Samples," *IEEE Trans Ind Appl*, vol. 56, no. 4, pp. 4415–4424, Jul. 2020, doi: 10.1109/TIA.2019.2958530.
[11] M. Aljabri *et al.*, "Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions," *IEEE Access*, vol. 10, pp. 121395–121417, 2022, doi: 10.1109/ACCESS.2022.3222307.
[12] J. Senanayake, H. Kalutarage, and M. O. Al-Kadri, "Android mobile malware detection using machine learning: A systematic review," *Electronics (Switzerland)*, vol. 10, no. 13. MDPI AG, Jul. 01, 2021. doi: 10.3390/electronics10131606.
[13] L. Gong *et al.*, "Systematically Landing Machine Learning onto Market-Scale Mobile Malware Detection," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 7, pp. 1615–1628, Jul. 2021, doi: 10.1109/TPDS.2020.3046092.
[14] M. Al-Kasassbeh, S. Mohammed, M. Alauthman, and A. Almomani, "Feature selection using a machine learning to classify a malware," in *Handbook of Computer Networks and Cyber Security: Principles and Paradigms*, Springer International Publishing, 2019, pp. 889–904. doi: 10.1007/978-3-030-22277-2_36.
[15] D. Gupta and R. Rani, "Improving malware detection using big data and ensemble learning," *Computers and Electrical Engineering*, vol. 86, Sep. 2020, doi: 10.1016/j.compeleceng.2020.106729.
[16] F. Xiao, Z. Lin, Y. Sun, and Y. Ma, "Malware Detection Based on Deep Learning of Behavior Graphs," *Math Probl Eng*, vol. 2019, 2019, doi: 10.1155/2019/8195395.
[17] S. Sharma, C. Rama Krishna, S. K. Sahay, and M. Scholar, "Detection of Advanced Malware by Machine Learning Techniques."
[18] I. Baptista, S. Shiaeles, and N. Kolokotronis, "A Novel Malware Detection System Based On Machine Learning and Binary Visualization."
[19] K. Lee, S. Y. Lee, and K. Yim, "Machine Learning Based File Entropy Analysis for Ransomware Detection in Backup Systems," *IEEE Access*, vol. 7, pp. 110205–110215, 2019, doi: 10.1109/ACCESS.2019.2931136.
[20] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A multimodal deep learning method for android malware detection using various features," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 773–788, Mar. 2019, doi: 10.1109/TIFS.2018.2866319.
[21] R. Kalakoti, S. Nomm, and H. Bahsi, "In-Depth Feature Selection for the Statistical Machine Learning-Based Botnet Detection in IoT Networks," *IEEE Access*, vol. 10, pp. 94518–94535, 2022, doi: 10.1109/ACCESS.2022.3204001.
[22] N. A. Azeez, O. E. Odufuwa, S. Misra, J. Oluranti, and R. Damaševičius, "Windows PE malware detection using ensemble learning," *Informatics*, vol. 8, no. 1, Mar. 2021, doi: 10.3390/informatics8010010.
[23] V. Kouliaridis and G. Kambourakis, "A comprehensive survey on machine learning techniques for android malware detection," *Information (Switzerland)*, vol. 12, no. 5, 2021, doi: 10.3390/info12050185.
[24] L. Cai, Y. Li, and Z. Xiong, "JOWMDroid: Android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters," *Comput Secur*, vol. 100, Jan. 2021, doi: 10.1016/j.cose.2020.102086.
[25] F. Khan, C. Ncube, L. K. Ramasamy, S. Kadry, and Y. Nam, "A Digital DNA Sequencing Engine for Ransomware Detection Using Machine Learning," *IEEE Access*, vol. 8, pp. 119710–119719, 2020, doi: 10.1109/ACCESS.2020.3003785.
[26] Y. Li, K. Xiong, T. Chin, and C. Hu, "A Machine Learning Framework for Domain Generation Algorithm-Based Malware Detection," *IEEE Access*, vol. 7, pp. 32765–32782, 2019, doi: 10.1109/ACCESS.2019.2891588.
[27] R. Kumar, X. Zhang, W. Wang, R. U. Khan, J. Kumar, and A. Sharif, "A Multimodal Malware Detection Technique for Android IoT Devices Using Various Features," *IEEE Access*, vol. 7, pp. 64411–64430, 2019, doi: 10.1109/ACCESS.2019.2916886.
[28] K. Shaukat, S. Luo, S. Chen, and D. Liu, "Cyber Threat Detection Using Machine Learning Techniques: A Performance Evaluation Perspective," in *1st Annual International Conference on Cyber Warfare and Security, ICCWS 2020 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020. doi: 10.1109/ICCWS48432.2020.9292388.
[29] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, and S. Venkatraman, "Robust Intelligent Malware Detection Using Deep Learning," *IEEE Access*, vol. 7, pp. 46717–46738, 2019, doi: 10.1109/ACCESS.2019.2906934.
[30] A. K. Ajay and J. C.D., "Automated multi-level malware detection system based on reconstructed semantic view of executables using machine learning techniques at VMM," *Future Generation Computer Systems*, vol. 79, pp. 431– 446, Feb. 2018, doi: 10.1016/j.future.2017.06.002.
[31] B. Urooj, M. A. Shah, C. Maple, M. K. Abbasi, and S. Riasat, "Malware Detection: A Framework for Reverse Engineered Android Applications Through Machine Learning Algorithms," *IEEE Access*, vol. 10, pp. 89031–89050, 2022, doi: 10.1109/ACCESS.2022.3149053.

[32] K. Bakour and H. M. Ünver, "VisDroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques," *Neural Comput Appl*, vol. 33, no. 8, pp. 3133–3153, Apr. 2021, doi: 10.1007/s00521-020-05195-w.
[33] J. Abawajy, A. Darem, and A. A. Alhashmi, "Feature subset selection for malware detection in smart iot platforms," *Sensors (Switzerland)*, vol. 21, no. 4, pp. 1–19, Feb. 2021, doi: 10.3390/s21041374.
[34] M. Ali, S. Shiaeles, G. Bendiab, and B. Ghita, "Malgra: Machine learning and N-GRAM malware feature extraction and detection system," *Electronics (Switzerland)*, vol. 9, no. 11, pp. 1–20, Nov. 2020, doi: 10.3390/electronics9111777.
[35] D. W. Fernando, N. Komninos, and T. Chen, "A Study on the Evolution of Ransomware Detection Using Machine Learning and Deep Learning Techniques," *Internet of Things*, vol. 1, no. 2. MDPI, pp. 551–604, Dec. 01, 2020. doi: 10.3390/iot1020030.
[36] H. Yang, S. Li, X. Wu, H. Lu, and W. Han, "A Novel Solutions for Malicious Code Detection and Family Clustering Based on Machine Learning," *IEEE Access*, vol. 7, pp. 148853–148860, 2019, doi: 10.1109/ACCESS.2019.2946482.
[37] H. El Merabet and A. Hajraoui, "A Survey of Malware Detection Techniques based on Machine Learning," 2019. [Online]. Available: www.ijacsa.thesai.org
[38] R. Feng *et al.*, "MobiDroid: A performance-sensitive malware detection system on mobile platform," in *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, Institute of Electrical and Electronics Engineers Inc., Nov. 2019, pp. 61–70. doi: 10.1109/ICECCS.2019.00014.
[39] W. Yuan, Y. Jiang, H. Li, and M. Cai, "A Lightweight On-Device Detection Method for Android Malware," *IEEE Trans Syst Man Cybern Syst*, vol. 51, no. 9, pp. 5600–5611, Sep. 2021, doi: 10.1109/TSMC.2019.2958382.
[40] H. Rathore, S. Agarwal, S. K. Sahay, and M. Sewak, "Malware Detection using Machine Learning and Deep Learning," Apr. 2019, doi: 10.1007/978-3-030-04780-1_28.
[41] Y. Gao, H. Hasegawa, Y. Yamaguchi, and H. Shimada, "Malware Detection by Control-Flow Graph Level Representation Learning With Graph Isomorphism Network," *IEEE Access*, vol. 10, pp. 111830–111841, 2022, doi: 10.1109/ACCESS.2022.3215267.
[42] W. K. Wong, F. H. Juwono, and C. Apriono, "Vision-Based Malware Detection: A Transfer Learning Approach Using Optimal ECOC-SVM Configuration," *IEEE Access*, vol. 9, pp. 159262–159270, 2021, doi: 10.1109/ACCESS.2021.3131713.
[43] X. Wang and C. Li, "Android malware detection through machine learning on kernel task structures," *Neurocomputing*, vol. 435, pp. 126–150, May 2021, doi: 10.1016/j.neucom.2020.12.088.
[44] H. M. Ünver and K. Bakour, "Android malware detection based on image-based features and machine learning techniques," *SN Appl Sci*, vol. 2, no. 7, Jul. 2020, doi: 10.1007/s42452-020-3132-2.
[45] J. Singh and J. Singh, "Detection of malicious software by analyzing the behavioral artifacts using machine learning algorithms," *Inf Softw Technol*, vol. 121, May 2020, doi: 10.1016/j.infsof.2020.106273.
[46] R. Damaševičius, A. Venčkauskas, J. Toldinas, and Š. Grigaliūnas, "Ensemble‐based classification using neural networks and machine learning models for windows pe malware detection," *Electronics (Switzerland)*, vol. 10, no. 4, pp. 1–26, Feb. 2021, doi: 10.3390/electronics10040485.
[47] A. Alotaibi, "Identifying Malicious Software Using Deep Residual Long-Short Term Memory," *IEEE Access*, vol. 7, pp. 163128–163137, 2019, doi: 10.1109/ACCESS.2019.2951751.
[48] K. Khariwal, J. Singh, and A. Arora, "IPDroid: Android malware detection using intents and permissions," in *Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability, WS4 2020*, Institute of Electrical and Electronics Engineers Inc., Jul. 2020, pp. 197–202. doi: 10.1109/WorldS450073.2020.9210414.
[49] E. Odat and Q. M. Yaseen, "A Novel Machine Learning Approach for Android Malware Detection Based on the Co- Existence of Features," *IEEE Access*, vol. 11, pp. 15471–15484, 2023, doi: 10.1109/ACCESS.2023.3244656.
[50] A. Azmoodeh, A. Dehghantanha, and K. K. R. Choo, "Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 88–95, Jan. 2019, doi: 10.1109/TSUSC.2018.2809665.

Authors' Profiles

Sadia Haq Tamanna completed her Bachelor of Science in Computer Science and Engineering at Bangladesh Army University of Engineering & Technology with a passion for delving into the intricacies of computer science. She is interested in subjects such as Machine Learning, Web development, software engineering, data structures, algorithm design, and Networking.

Muhammad Muhtasim is currently working as a Lecturer in the Department of Computer Science and Engineering at Bangladesh Army University of Engineering & Technology (BAUET). He completed his bachelor's degree in computer science and engineering and master's in computer science and information Technology from Patuakhali Science and Technology University. He is interested in research areas including Data Mining, Machine Learning, Data Analytics, Data Mining and Cyber Security.

Aroni Saha Prapty is currently working as a Lecturer in the Department of Computer Science and Engineering at Bangladesh Army University of Engineering & Technology (BAUET). She completed her bachelor's degree in computer science and engineering from Khulna University of Engineering & Technology. She is interested in research areas including Machine Learning and Natural Language processing.

Amrin Nahar completed her Bachelor of Science in Computer Science and Engineering at Bangladesh Army University of Engineering & Technology with a passion for delving into the intricacies of computer science. She is interested in subjects such as Machine Learning, Web development, software engineering, data structures, algorithm design, and Networking.

Md. Tanvir Ahmed Tagim completed his Bachelor of Science in Computer Science and Engineering at Bangladesh Army University of Engineering & Technology with a passion for delving into the intricacies of computer science. He is interested in subjects such as Machine Learning, Web development, software engineering, data structures, algorithm design, and Networking.

Fahmida Rahman Moumi completed her Bachelor of Science in Computer Science and Engineering at Bangladesh Army University of Engineering & Technology with a passion for delving into the intricacies of computer science. She is interested in subjects such as Machine Learning, Web development, software engineering, data structures, algorithm design, and Networking.

Shadia Afrin completed her Bachelor of Science in Computer Science and Engineering at Bangladesh Army University of Engineering & Technology with a passion for delving into the intricacies of computer science. She is interested in subjects such as Machine Learning, Web development, software engineering, data structures, algorithm design, and Networking.

How to cite this paper: Sadia Haq Tamanna, Muhammad Muhtasim, Aroni Saha Prapty, Amrin Nahar, Md. Tanvir Ahmed Tagim, Fahmida Rahman Moumi, Shadia Afrin, "Evaluation of Machine Learning Algorithms for Malware Detection: A Comprehensive Review", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.15, No.2, pp. 51-67, 2025. DOI:10.5815/ijwmt.2025.02.05